Virtualización

José M. Peña

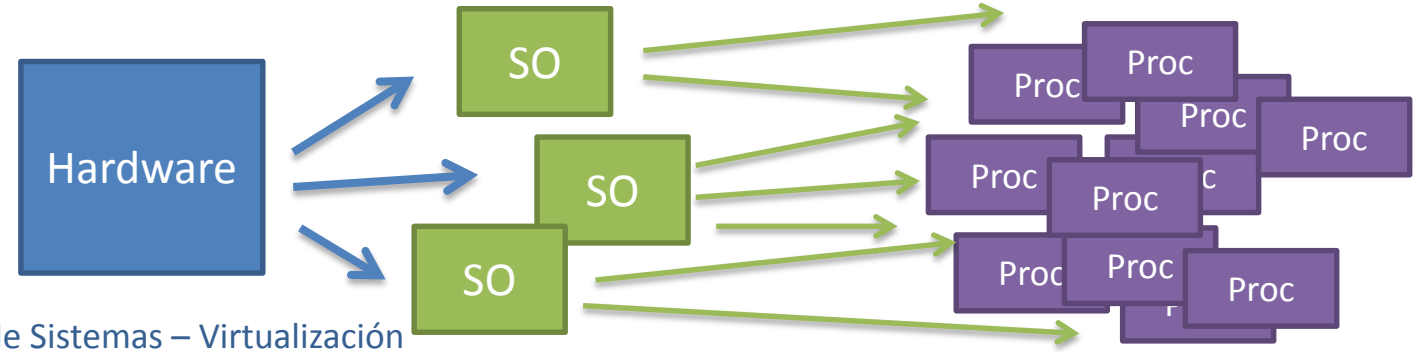
<jmpena@fi.upm.es>

Contenidos

1. Conceptos:
   - Definiciones.
   - Requisitos y ventajas.
2. Técnicas de virtualización:
   - Virtualización completa.
   - Para-virtualización.
   - Virtualización asistida por hardware
3. Virtualización cliente/servidor.

Conceptos Previos

- Aislamiento entre el hardware y las aplicaciones y entre las aplicaciones entre sí:
  - Sistema operativo: Aísla un proceso de otro.
    - ¿Qué aísla dos instalaciones de sistema operativo entre sí?
    - A efectos de incompatibilidades en versiones de librería instaladas.
    - Configuraciones específicas de servicios.
  - Sistema operativo: Gestiona el hardware para varios procesos.
    - La capacidad de aprovechamiento de ese hardware se ajusta al compromiso de las aplicaciones que corren sobre él.

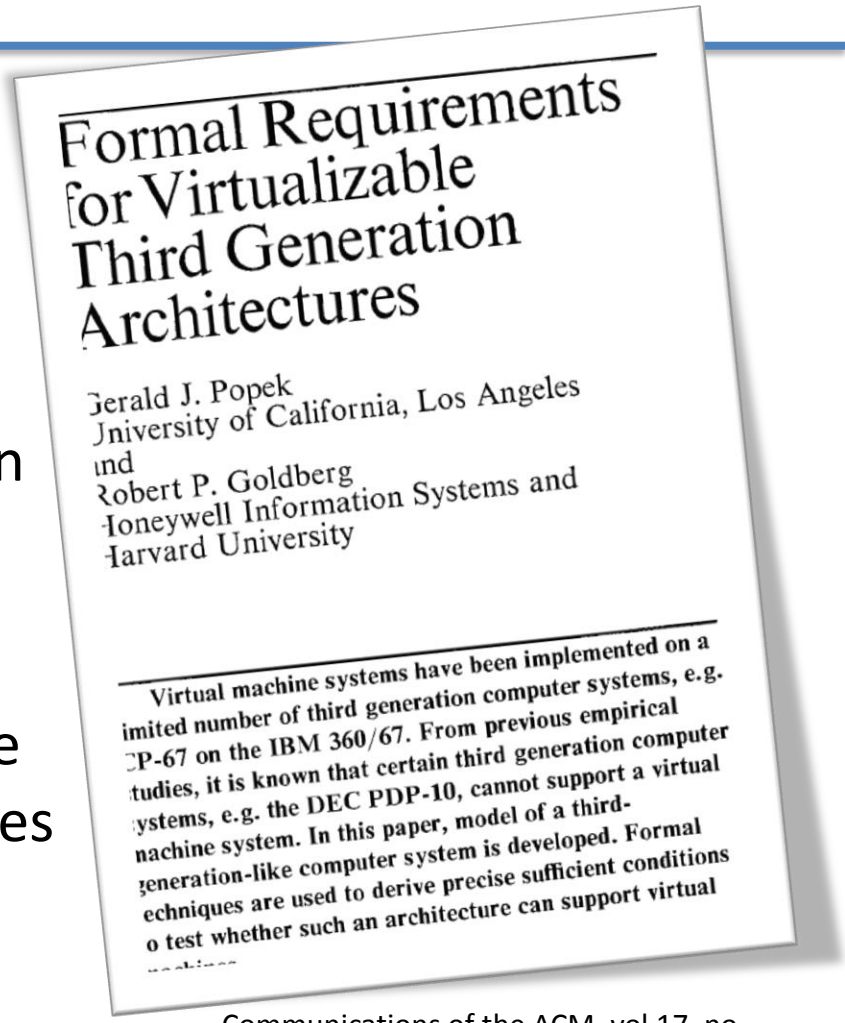


Definición de Virtualización

- Proyección de una serie de recursos o servicios reales del sistema hacia un subsistema de forma que sólo una parte de ellos sean visibles para el mismo.
- La visibilidad también restringe lo que el subsistema puede hacer con esos recursos/servicios.
- Dicha visibilidad está definida por una interfaz de uso (que puede ser muy parecida a la interfaz nativa de esos recursos/servicios)

Definición de Virtualización

- Definición formal:
  - La virtualización implica la construcción de un isomorfismo que proyecta un sistema virtual alojado sobre un sistema anfitrión real (Popek and Goldberg, 1974)
- Definición para sistemas:
  - Particionamiento del hardware de un sistema y creación de diferentes entornos virtuales (máquinas virtuales) que pueden albergar subsistemas independientes.



Formal Requirements for Virtualizable Third Generation Architectures

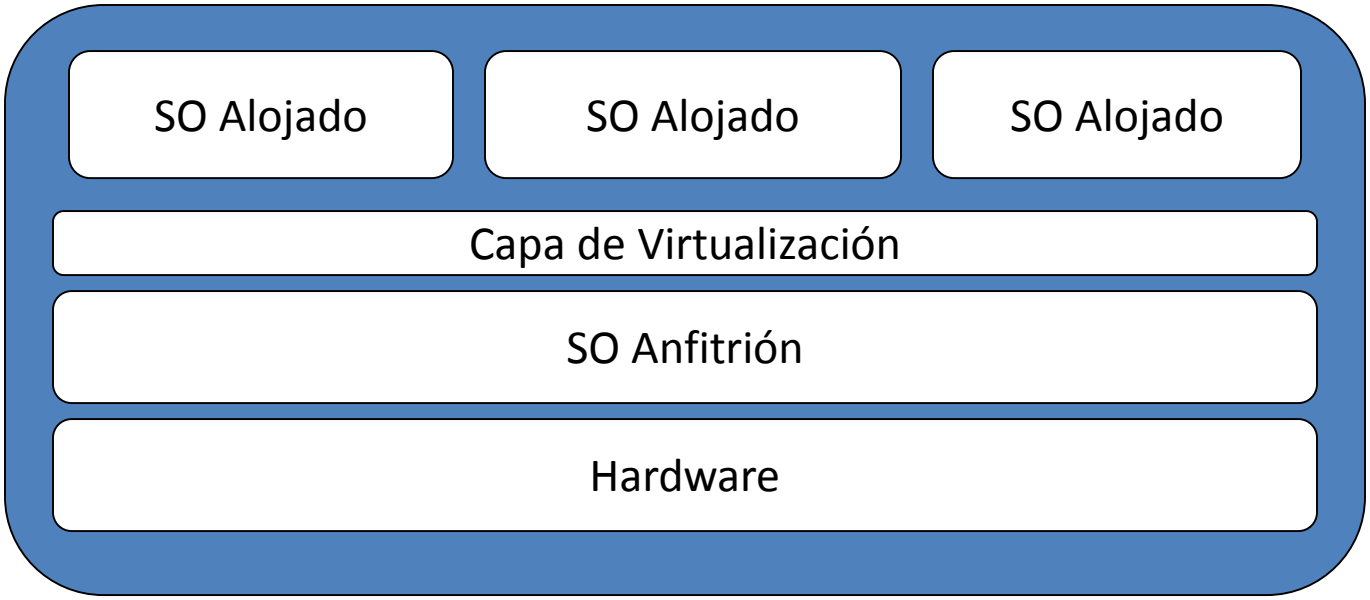
Gerald J. Popek
University of California, Los Angeles
and
Robert P. Goldberg
Honeywell Information Systems and
Harvard University

Virtual machine systems have been implemented on a limited number of third generation computer systems, e.g. CP-67 on the IBM 360/67. From previous empirical studies, it is known that certain third generation computer systems, e.g. the DEC PDP-10, cannot support a virtual machine system. In this paper, model of a third-generation-like computer system is developed. Formal techniques are used to derive precise sufficient conditions to test whether such an architecture can support virtual

Communications of the ACM, vol 17, no 7, 1974, pp.412-421

Virtualización Genérica en Sistemas

- Virtualización del Sistema Operativo (Máquinas Virtuales)
  - Separación entre SO y el hardware real.
  - SO Anfitrión (Host)→ SO Alojado (Guest)



Consideraciones Importantes

- Eficiencia:
  - Aquellas instrucciones inocuas deben ejecutarse sobre el hardware real (si es compatible).
  - Diferente en el caso de una emulación.
- Control de recursos:
  - Los programas en ejecución sólo pueden consumir los recursos otorgados al subsistema donde ejecutan.
- Equivalencia:
  - Los resultados de la ejecución sobre el sistema nativo y sobre el sistema virtualizado deben ser equivalentes.
  - A diferencia, únicamente, de aspectos de temporización o consumo de recursos.

Ventajas de la Virtualización

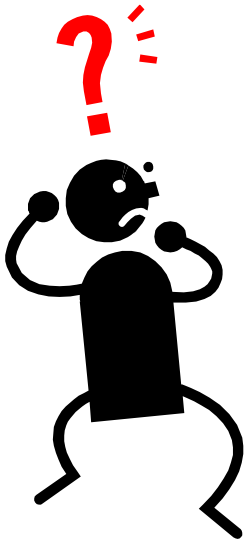
- Despliegue de sistemas:
  - Se necesita validar sólo una plataforma.
  - Se virtualiza dicha plataforma sobre el hardware que sea.
  - Mayor facilidad de encapsulación (migración y replicación).
- Eficiencia y reutilización:
  - Mayor aprovechamiento del hardware.
  - Escalabilidad y reparto de recursos.
- En estaciones de trabajo:
  - Coexistencia de varios SO en el mismo equipo.
  - Validación multiplataforma o aplicaciones dependientes de SO.
  - Simulación de una red (heterogénea) de equipos.

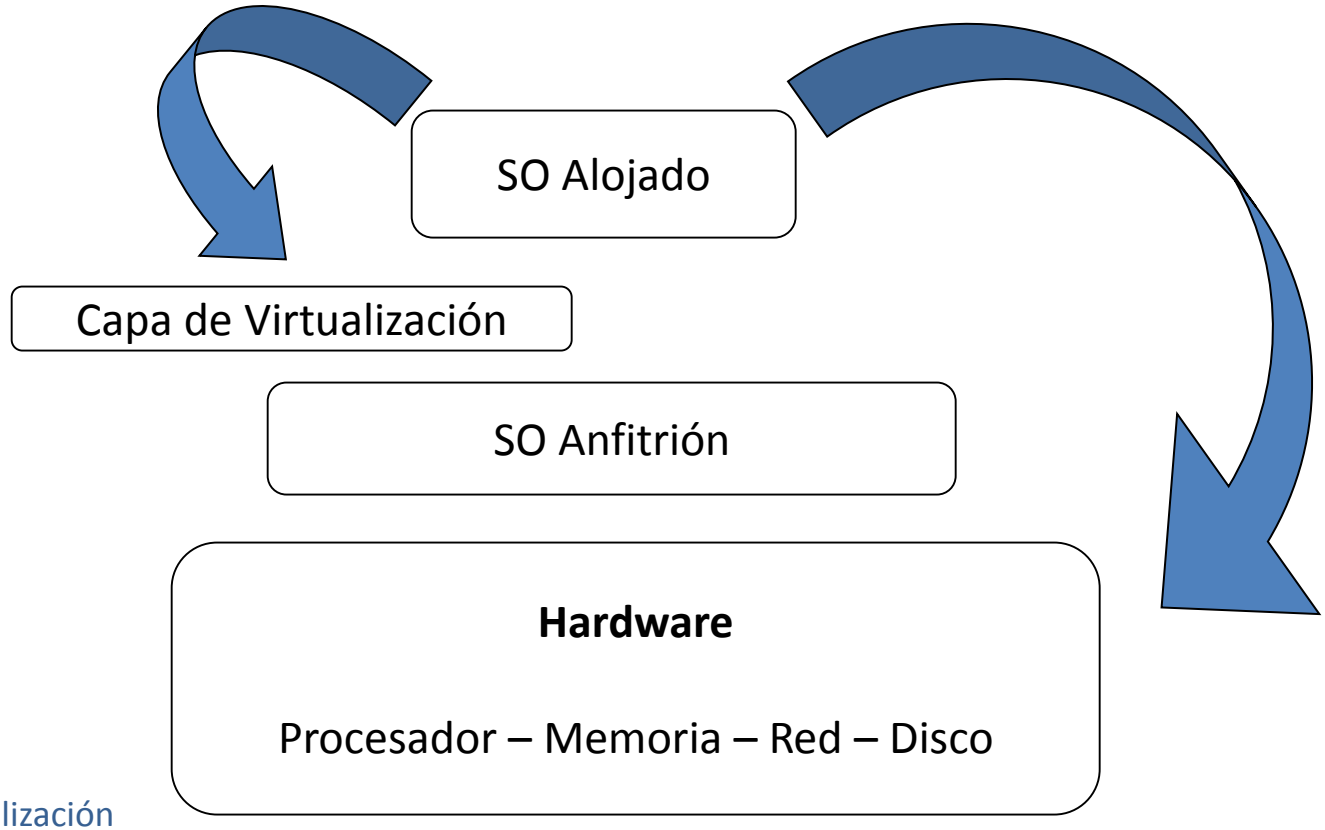
Ventajas de la Virtualización

- Seguridad:
  - Aísla aplicaciones “peligrosas”.
  - Concepto de *sandbox* asociado a la MV.
  - Separación entre SO de otras MV y del SO anfitrión.
  - Particionamiento de servidores (*hosting*): por incompatibilidades o por estabilidad del entorno.
  - Aplicaciones de tipo *legacy*.
- Desarrollo de software en el núcleo del SO:
  - Muy difícil depuración en SO nativo.
  - Si “casca” el SO alojado no afecta al SO anfitrión.
  - Facilidades para hacer volcado de registros, memoria, puntos de interrupción.

Técnicas de Virtualización

- ¿De qué manera el SO Alojado accede al hardware?
- ¿El SO alojado debe ser consciente de que está virtualizado (vale un SO tradicional)?





Principios sobre Virtualización

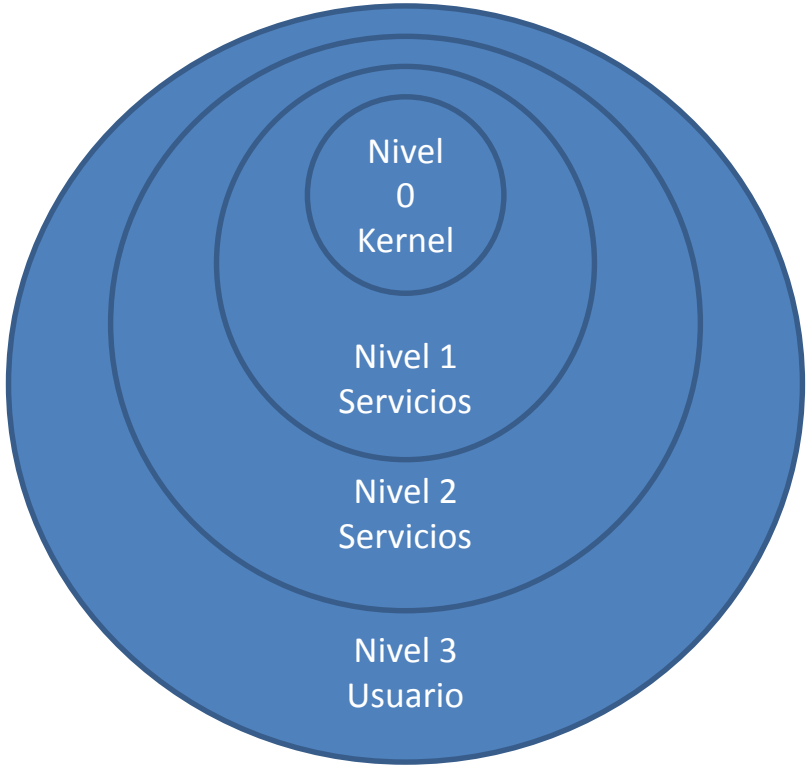
- La virtualización está ligada al tipo de instrucciones (máquina) y a los recursos que usan:
  - Instrucciones privilegiadas: Instrucciones sólo ejecutables en modo privilegiado.
  - Instrucciones sensibles:
    - A nivel de control: Modificaciones de la ubicación de memoria o de los modos de privilegio.
    - A nivel de comportamiento:
      - Dependientes de localización: El comportamiento depende de su localización en memoria.
      - Dependientes de modo: El comportamiento depende del modo de privilegio.
  - Inocuas: Instrucciones no sensibles

Un sistema de virtualización se puede construir sólo cuando las instrucciones sensibles son un subconjunto de las instrucciones privilegiadas.

Niveles de Privilegio (ejemplo x86)

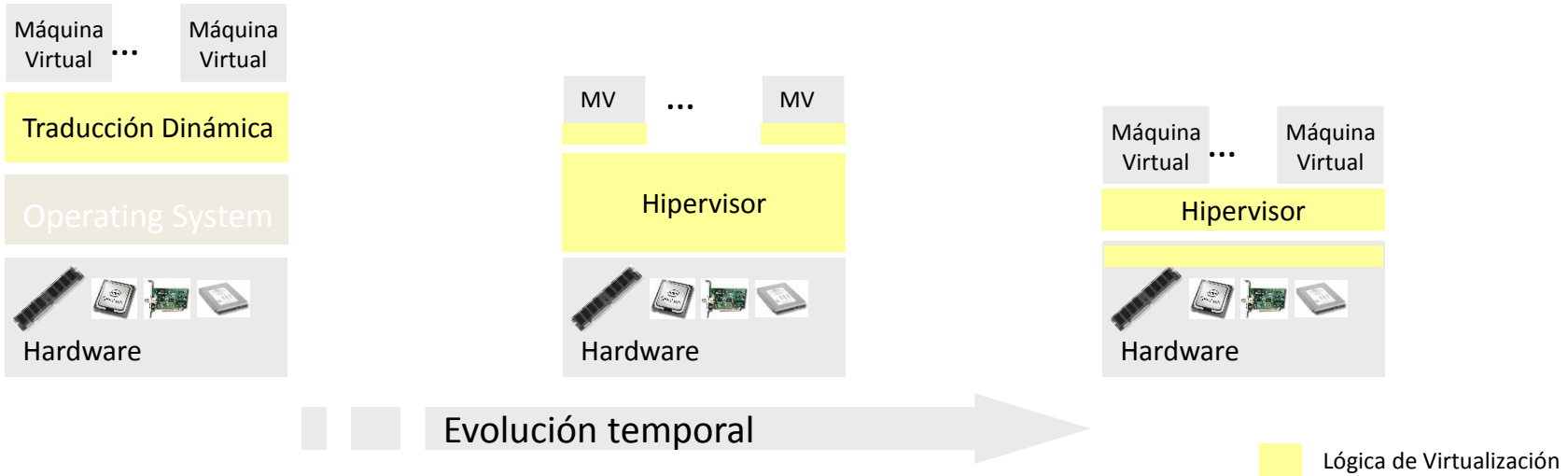
- Los procesadores x86 disponen de 2 bits para indicar el nivel de privilegio (4 posibles niveles).
  - Típicamente los sistemas operativos (e.g., Linux) sólo usan 2 de ellos.

- Los registros de segmento (de código o de pila) contiene los valores de estos bits.
  - Son instrucciones privilegiadas las operaciones que cargan valores sobre estos registros.
- También se incluye esta información en las tablas de traducción de páginas.



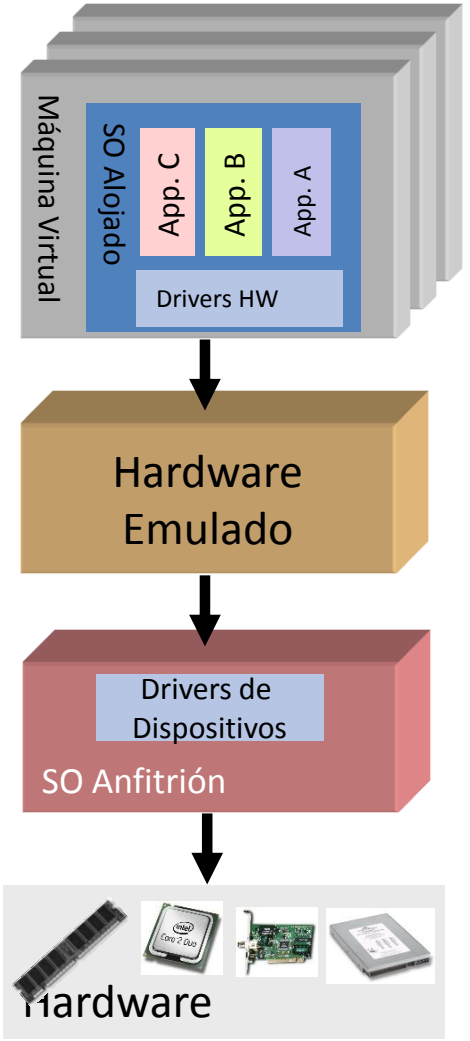
Alternativas de Virtualización

- 1ª Generación: Virtualización completa
  - Re-implementación binaria del hardware
  - Basada 100% en capas software.
- 2ª Generación: Para-virtualización
  - Virtualización cooperativa
  - SO alojados modificados
- 3ª Generación: Virtualización asistida por el hardware.
  - No se altera el SO alojado
  - Plataformas hardware diseñadas para soportar virtualización



Virtualización Completa

- 1ª Generación de virtualización para servidores
- Traducción binaria dinámica
  - El nivel de emulación implementa un hardware virtual que le solicitará determinadas operaciones al hardware real si lo necesita.
  - El SO alojado no sabe que se está ejecutando sobre un entorno virtual.
- Todo el hardware es emulado (incluida la CPU).
- Eso implica que puede ser distinto del hardware real.

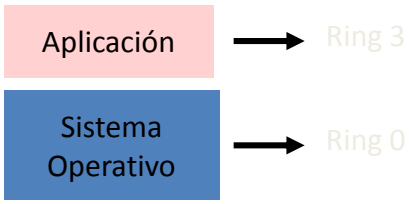


Virtualización Completa – Ventajas

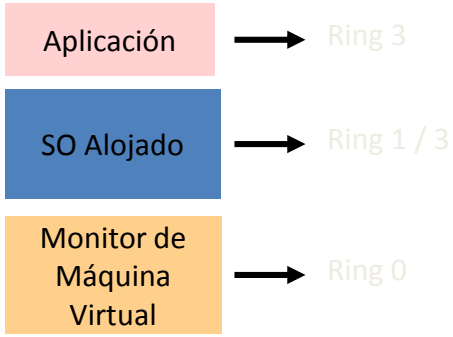
- El nivel de emulación:
  - Aísla las MV entre sí y del SO anfitrión.
  - Controla el acceso de las MV a los recursos del sistema y previene que MV inestables/insegures afecten al sistema.
- Portabilidad completa de MVs:
  - Por medio de la emulación de un conjunto consistente de hardware emulado (independiente del hardware real). Las MVs tiene la posibilidad de moverse, de forma transparente, entre sistemas con hardware diferente sin problemas.
    - Es posible ejecutar una aplicación o SO que fue desarrollado para otra arquitectura diferente del hardware real.
    - Usado en el desarrollo sobre hardware experimental o específico.

Virtualización Completa - Inconvenientes

- La emulación de hardware tiene un coste en rendimiento.
- Los núcleos de los SOs están diseñados con la idea de correr en modo privilegiado.
  - El uso de ciertas instrucciones sobre registros o memoria está pensadas para agilizar ciertas operaciones. Cosa que no ocurre si implica en trap a nivel privilegiado en cada caso.
- El rendimiento motivó nuevas aproximaciones.



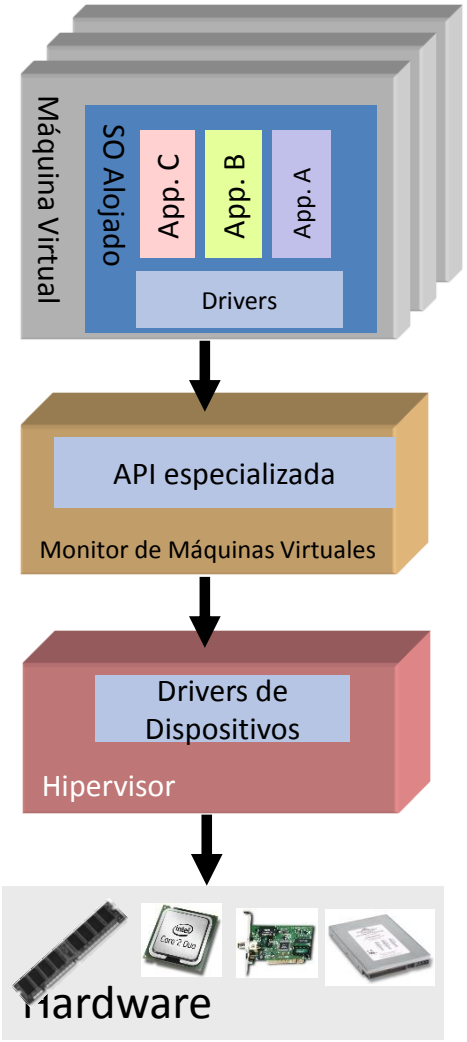
Arquitectura Tradicional



Virtualización Completa

Para-Virtualización

- Los SOs alojados se modifican de forma que su ejecución en modo no-privilegiado sea eficiente:
  - El SO alojado es completamente cosciente de que está virtualizado y debe saber como implementar las operacioes que requieren modo privilegiado.
  - Por lo tanto, el monitor de MVs no tiene que traducir instrucciones privilegiadas.
  - El SO alojado usa un API especializada para dialogar con el Monitor de MVs que le da funciones que sustituyen a las instrucciones privilegiadas.
- El Monitor de MVs es responsable de manejar las solicitudes virtualizadas y pasárselas al hardware.

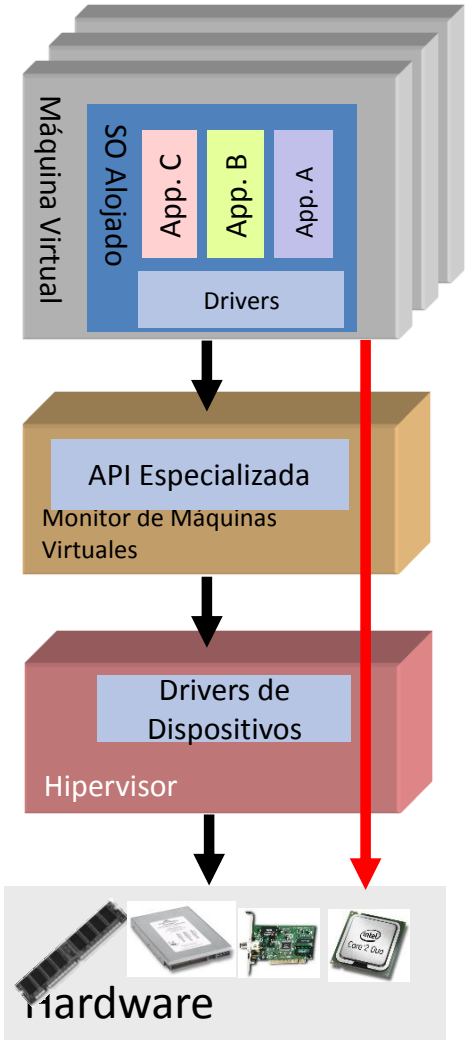


Para-Virtualization

- Los SO alojados se modifican de dos diferentes maneras:
  - Recompilando el núcleo del SO alojado:
    - El SO debe sustituir determinados fragmentos de código por llamadas al API especializada.
    - Como el SO se ve modificado y resulta necesario recompilarlo entro (se pueden distribuir versiones recompiladas con esta opción):
      - Algunos fabricantes (como Novell) distribuyen versiones recompiladas para paravirtualizción, otros (como Microsoft) no.
  - Instalación de drivers para paravirtualización:
    - En el caso de algunos SOs no es posible una paravirtualización completa (que requiere modificar y recompilar el SO).
    - Para asegurarse un buen rendimiento (en comparación con la virtualización completa), sólo se paravirtualizan ciertos dispositivos.
    - Por ejemplo, las instrucciones asocidas a tarjetas gráficas o de red pueden modificarse o no antes de que salgan de la máquina virtual por unos driver específicos para ello.

Virtualización Asistida por el Hardware

- El SO alojado corre en modo privilegiado.
- El monitor de MVs usa extenciones (del tipo Intel®-VT or AMD-V®) para interceptar e interpretar ciertas operaciones privilegiadas.
- La virtualización de este tipo elimina gran parte de las complicaciones de escribir un monitor de MVs.
- El monitor de MVs tiene determinadas instrucciones “super-privilegiadas”.



Soporte en Chip (x86) de Virtualización

Empieza a ser implantada desde las versiones de 64Bits.

- AMD-V: AMD Virtualization (SVM- Secure Virtual Machine):
  - Últimos Opteron y Phenom II (dan soporte a segunda generación)
  - Soporte para virtualización de tablas de páginas (Rapid Virtualizaton Indexing)

- VT-x: Intel Virtualization:
  - Ultima versión desde Nehalem.
  - Extended Page Tables (mismas ideas de RVI de AMD).

Virtualización Asistida por el Hardware

- Ventajas
  - Permite ejecutar versiones no modificadas de SO alojados (SO antiguos pueden ejecutarse sin problemas).
- Inconvenientes
  - Velocidad y Flexibilidad
    - Aunque las instrucciones privilegiadas se pueden ejecutar (por pare del SO alojado), si se supiese que se ejecuta en un entorno virtual eso simplificaría (y agilizaría) determinadas operaciones.
      - Una alernativa razonable es usar algo parecido a una paravirtualización puntual.

Implementaciones de Virtualización

Virtualización completa:

- VMWare Workstation (versión de escritorio)
- z/VM de IBM (entornos mainframe)
- Boch (emulador de x86)
- QEMU (emulador de diversas arch.)

Para-virtualización:

- User-mode Linux (Linux sobre Linux, para desarrollo)
- Xen (multi SO)

Asistida por hardware:

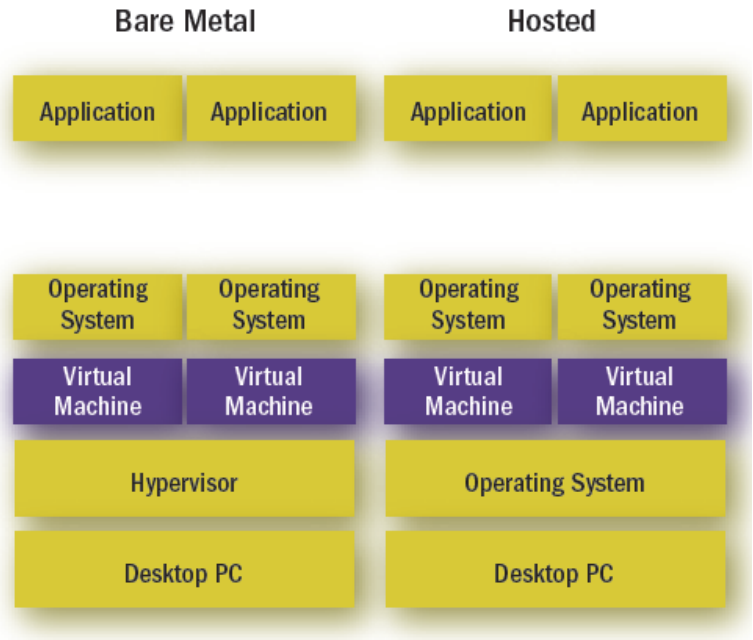
- AMD-V
- Intel VT-x

Virtualización Cliente/Servidor

- Diferentes alternativas de organizar la virtualización en una arquitectura de tipo cliente/servidor:
  - Virtualización en cliente.
  - Virtualización de *workspaces*:
    - En servidor.
    - En cliente.
  - Aislamiento de aplicaciones:
    - *Stream* de aplicaciones.

Virtualización en Cliente

- Monitor de MVs o hipervisor corriendo en la máquina cliente.
- Ejemplos:
  - Microsoft Virtual PC
  - Parallels Desktop for Mac
  - VMware Fusion
  - WINE.
- Casos de uso:
  - Ejecución de aplicaciones Windows sobre Mac,
  - Pruebas de código dentro de MVs
  - Emulación de antiguo hardware de videojuegos

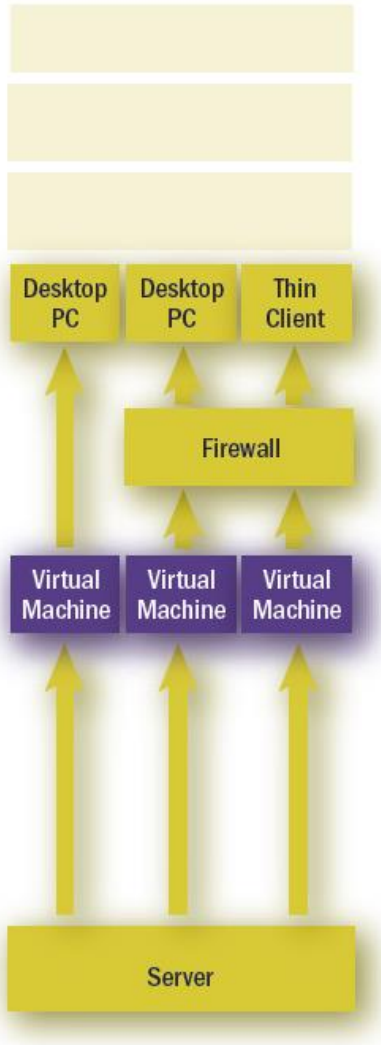


- Los entornos virtuales en cliente, no escalan necesariamente para proporcionar soluciones corporativas.
- Determinados objetivos se alcanzan por medio de la virtualización en servidor (productos específicos).

Virtualización de *Workspaces* en Servidor

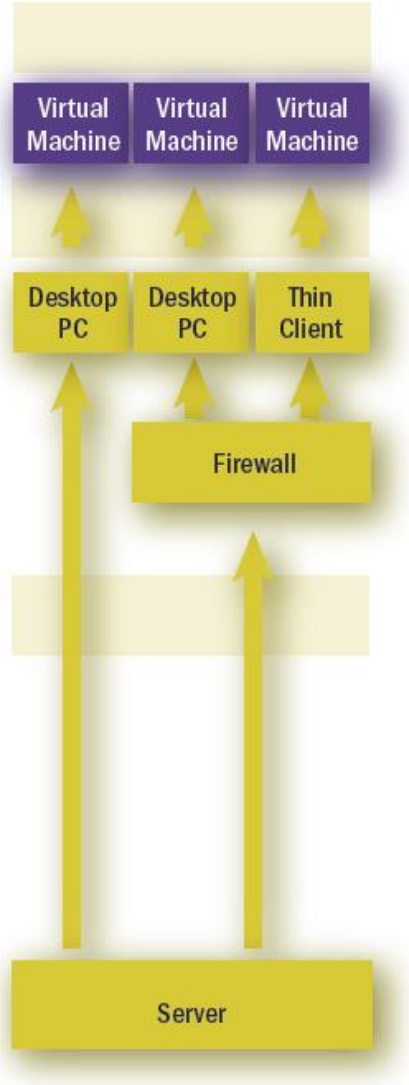
- Un *workspace* (un escritorio de sistema operativo con configuración personalizada) que se ejecuta dentro de una máquina virtual alojada en el **servidor**.
- Ejemplos:
  - VMware VDI
- Casos de uso:
  - Infraestructura de escritorios gestionada centralmente.
  - Garantización de entornos seguros y unificados.

- Un repositorio de *workspaces* virtuales que residen en el servidor. Los usuarios acceden al sistema por medio de terminales remotos (e.g., Microsoft's Remote Desktop Protocol – RDP)
- Es posible personalizar entornos de usuarios mantenido unas facilidades de administración razonables.
- Es posible tener *brokers* (e.g., Propero) intermedios que equilibren la carga frente a un pool de servidor de *wokspaces*.
- El principal problema de la virtualización de *workspaces* en servidor es que es un consumidor voraz de ancho de banda. El rendimiento está sujeto a la red.



Virtualización de *Workspaces* en Cliente

- Un *workspace* (un escritorio de sistema operativo con configuración personalizada) que se ejecuta dentro de una máquina virtual alojada en el **cliente**.
- Ejemplos:
  - Kidaro Managed Workspace
  - Sentillion vThere
- Casos de uso:
  - Acceso remoto seguro
  - Protección de datos sensibles (en entornos d defensa o salud).
  - Ordenadores particulares ejecutando de forma remota un escritorio corporativo.

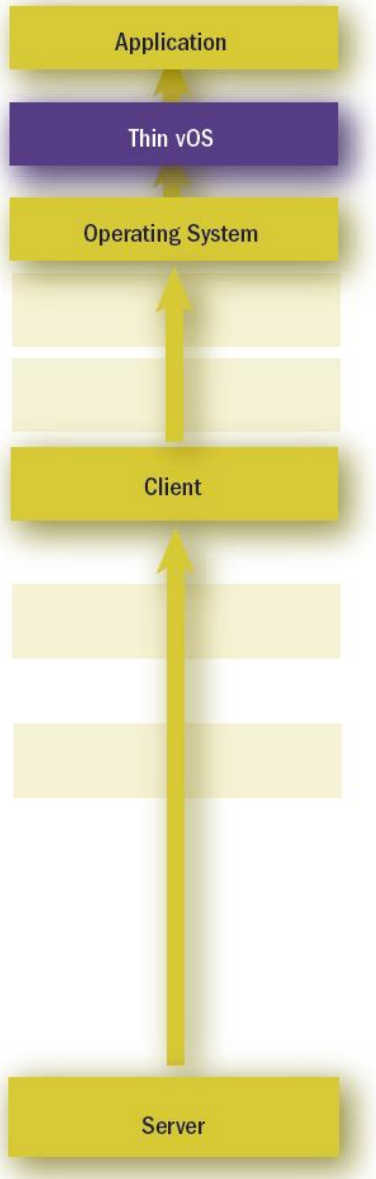


- El *workspace* virtual se sirve de forma remota para que lo ejecute el cliente.
- Centraliza la gestión y reparte la carga.
- La principal ventaja de esta alternativa es que proporcional seguridad y aislamiento de los datos y lógica de control en el cliente.
- Es el modelo ideal para organizaciones que necesitan dar seguridad a entornos servidor a usuarios remotos.

Aislamiento de Aplicaciones

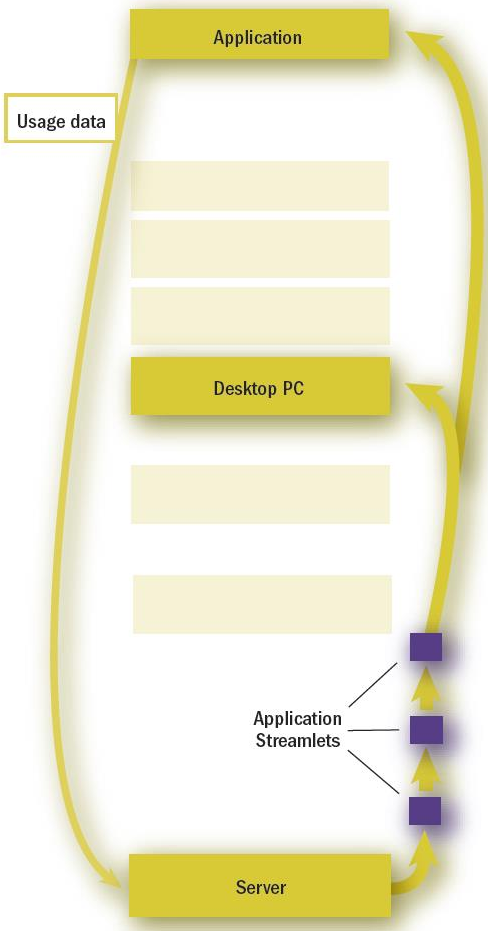
- Una aplicación empaquetada con su copia virtual del SO (de forma que aspectos de configuración, etc. se pueden obviar).
- Ejemplos:
  - Thinstall
  - Trigence
- Casos de uso:
  - Sobrevivir al infierno de las DLLs
  - Creación de *sandboxes* para la ejecución segura de aplicaciones.

- Las aplicaciones usan un registro virtual (Thinstall) y un sistema de ficheros, todo ello embebido en el paquete en el que viene la aplicación
  - Estos contenidos adicionales preservan de cambios y problemas de incompatibilidad con el entorno de escritorio existente.
- Principalmente para Windows, en menor medida en Linux o Solaris.
- Pega: Incrementa el tamaño de los paquetes de distribución de aplicaciones y unos mayores requisitos de recursos (disco/memoria).



Streaming de Aplicaciones

- Despliegue *just-in-time* de aplicaciones almacenadas en el servidor a los clientes, de forma que las aplicaciones puedan empezar a ejecutar antes de que el entorno completo se haya descargado.
- Ejemplos:
  - AppStream
  - Microsoft SoftGrid
- Casos de uso:
  - Gestión de un número de instancias de aplicaciones en ejecución que estén sujetas a un *pool* de licencias.



- Subtipo de “Aislamiento de Aplicaciones”que incluye un mecanismo especializado de despliegue y de ejecución.
  - Mandas en un *stream* el código de aplicación al cliente donde ejecutará aislado.
- No se suele hacer con entornos de escritorio completos, sólo la aplicación, de forma que se debe proveer de un *workspace*:
  - Requiere mantener el SO en el cliente y asegurar la compatibilidad.

Referencias

- Amit Sing, An Introduction to Virtualization
  - http://www.kernelthread.com/publications/virtualization/
- Vmware, Virtualization Overview
  - http://www.vmware.com/virtualization/
- Vmware, Virtualization White Paper
  - http://www.vmware.com/pdf/virtualization.pdf [pdf]
- Cambridge University Systems Research Group, Xen Website
  - http://www.cl.cam.ac.uk/Research/SRG/netos/xen/
- Microsoft, Virtual PC Website
  - http://www.microsoft.com/windows/virtualpc/default.mspx
- Bochs Website
  - http://bochs.sourceforge.net/
- Intel Virtualization Technology
  - http://www.intel.com/technology/virtualization/
- AMD, "Pacifica" Virtualization Technology
  - http://enterprise.amd.com/Downloads/Pacifica_en.pdf [pdf]
- James Smith, Ravi Nair, "The Architectures of Virtual Machines," IEEE Computer, May 2005, pp. 32-38.
- Mendel Rosenblum, Tal Garfinkel, "Virtual Machine Monitors: Current Technology and Future Trends," IEEE Computer, May 2005, pp. 39-47.
- L.H. Seawright, R.A. MacKinnon, "VM/370 – a study of multiplicity and usefulness," IBM Systems Journal, vol. 18, no. 1, 1979, pp. 4-17.
- S.T. King, G.W. Dunlap, P.M. Chen, "Operating System Support for Virtual Machines," Proceedings of the 2003 USENIX Technical Conference, June 9-14, 2003, San Antonio TX, pp. 71-84.
- A. Whitaker, R.S. Cox, M. Shaw, S.D. Gribble, "Rethinking the Design of Virtual Machine Monitors," IEEE Computer, May 2005, pp. 57-62.
- G.J. Popek, and R.P. Goldberg, "Formal requirements for virtualizable third generation architectures," CACM, vol. 17 no. 7, 1974, pp. 412-421.